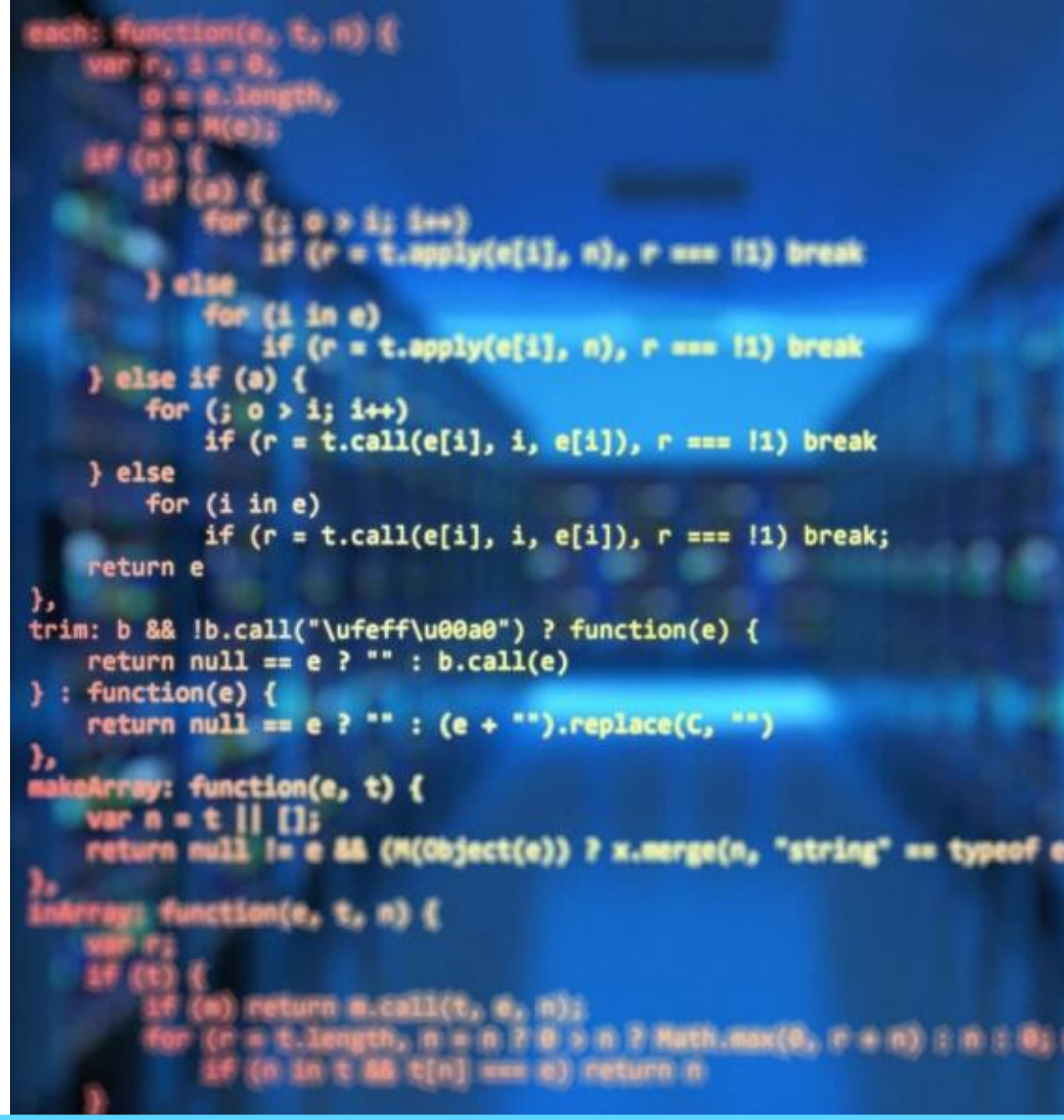
ПРОГРАММА КУРСА «ВЕБ-МАСТЕР»

Тема 1.3 Изображения на веб-странице



До этого момента мы с тобой рассматривали только работу с текстом. Но разве можно себе представить настоящий яркий сайт, на котором совсем нет картинок? Совершенно невозможно!

В этом уроке ты научишься вставлять на веб-страницу изображения: картинки, фотографии, векторные рисунки. А также узнаешь, что такое карты изображений с активными областями!

В самом начале, когда мы просто рассуждали о том, что такое язык гипертекстовой разметки, ты на рассматриваемом примере мог убедиться, что для вставки изображений на страницу существует специальный тег, который указывает браузеру, в каком месте документа необходимо вставить изображение и из какого файла.

Если помнишь, это тег <img>. Этот тег не является парным, поэтому вся информация о вставляемом изображении прописывается в его атрибутах.

Атрибут **src** является обязательным, он указывает браузеру расположение графического файла, который необходимо поместить на страницу. При этом в качестве значения этого атрибута можно указать как абсолютный адрес файла, так и относительный.

Абсолютный адрес файла — это полный путь к нему. Например, если файл logo.jpg находится на сайте www.site.ru, да ещё и в папке images, то абсолютный путь к нему будет следующий:

http://www.site.ru/images/logo.jpg.

Абсолютные пути используются крайне редко, это бессмысленно, так как абсолютный путь заставляет браузер делать больше работы, запрашивая каждый раз IP-адрес изображения от DNS-сервера.

Гораздо удобнее указать путь к файлу относительно текущей страницы, ведь почти всегда изображения для сайта размещаются на том же сервере, что и html-документ.
В этом случае указывается местоположение графического файла относительно расположения веб-странички. Например, если html-страница и картинка logo.jpg размещены в одной папке, то для вставки этой картинки на страницу нужно задать следующее значение атрибуту src:

```
<img src="logo.jpg">
```

Как правило, на сайте размещены сотни картинок, хранить их в одной папке с html-страницами не очень удобно. Поэтому в корневом каталоге сайта обычно создают папку специально для графических файлов — images (или img) — и помещают в неё все изображения, а путь к графическим файлам в этом случае прописывается так:

```
<img src="images/logo.jpg">
```

<img src="images/logo.jpg">

Это в том случае, когда html-страница находится в одном каталоге с папкой **images**. Если же html-документ и папка с графическими файлами находятся в разных каталогах, то перед images в адресе добавляется запись **../** — означает подняться вверх на один уровень, к корневому каталогу.

Кстати, поисковые системы читают имена изображений и считают их для оптимизации поискового запроса. Поэтому рекомендуется присваивать изображениям смысловые имена: logo.jpg лучше, чем img1.jpg.

Все остальные атрибуты тега <img> не являются обязательными и используются по мере необходимости.

Атрибуты **width** и **height** — ширина и высота изображения в пикселях. В этих атрибутах чаще всего указываются реальные размеры изображения, их наличие не обязательно, но считается признаком хорошего тона веб-мастера.

С чем это связано?

1

Дело в том, что текст передаётся по каналам связи гораздо быстрее, чем графика, поэтому и отображается на странице раньше картинок. Но если браузер начал отображать страницу, а сам ещё не знает, какого размера картинка размещена на ней, то он оставляет под изображение совсем мало места. В результате во время загрузки страницы в браузере можно наблюдать ситуацию, когда текст несколько раз «переезжает» по мере того, как подгружается графика. А вот если браузер заранее знает, какого размера картинка (это указано в атрибутах), то он «забронирует» под картинку необходимое пространство, и никакие элементы сайта никуда не уедут.

2

Если будет задан только один из атрибутов, то второй будет вычисляться автоматическ и для сохранения пропорций рисунка.

3

Если указанные размеры изображения больше реальных, то браузер растянет картинку до указанных размеров, если меньше — сожмёт. Но для изменения размера изображения лучше всё-таки пользоваться графическим редактором.

Атрибут **alt** позволяет задать описание картинки для тех пользователей, у которых в браузере отключён показ графики. В этом случае браузер зарезервирует место под изображение, но вместо самой картинки покажет текст, который указан в качестве значения этого атрибута.

Альтернативный текст выведется и в том случае, если произошло что-то, в результате чего изображение просто не отобразилось. А речевые браузеры, которые предназначены для пользователей со значительным нарушением зрения, умеют читают альтернативный текст вслух.

Не следует использовать alt, если можно обойтись без него, другими словами, если пользователи ничего не потеряют, когда изображение не появится.

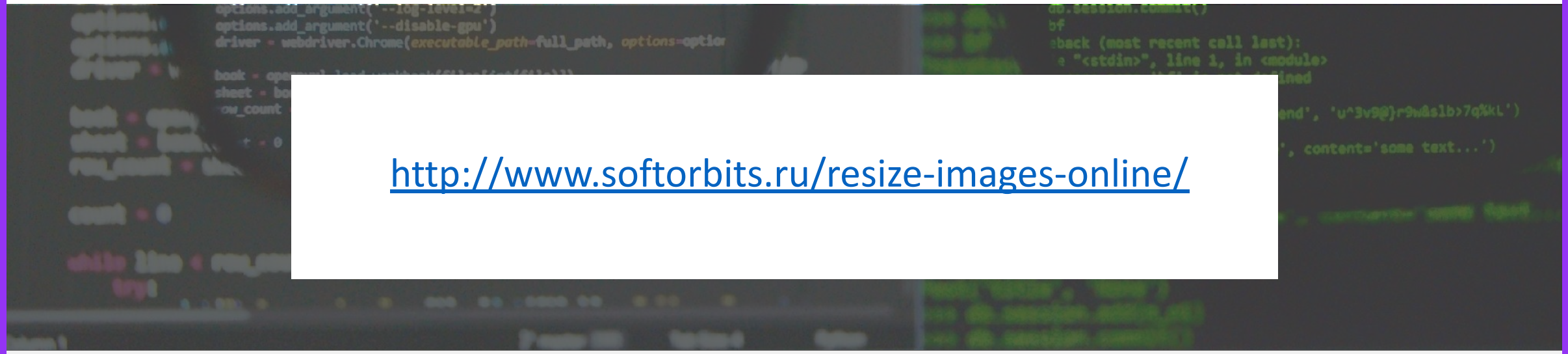
Теперь давай разберёмся со вставкой изображения на практике. На страницу, созданную в предыдущем уроке, поместим свою фотографию.

Фотографии хорошего качества обычно имеют достаточно большой размер в пикселях, нам же достаточно будет изображения не больше 300 пикселей в ширину. Если твоё изображение слишком велико, то его лучше уменьшить с помощью графического редактора (например, Paint) или онлайн-конвертера.

Даже если ты ни разу не работал с графическим редактором, ты быстро разберёшься, как изменить в нём размеры изображения. Информацию о том, как сделать это именно в том графическом редакторе, который оказался у тебя под рукой, ты всегда можешь найти в сети Интернет. Просто воспользуйся любой поисковой системой.

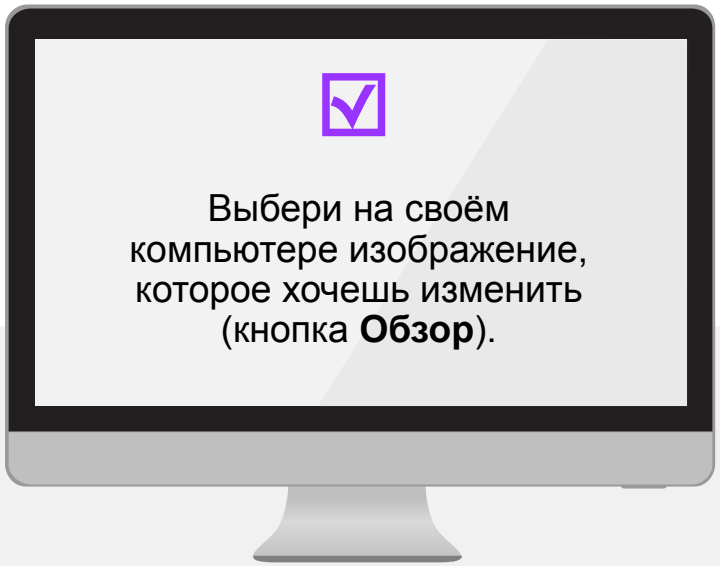
Но есть ещё более простой и удобный способ — онлайн-конвертер! Их на самом деле много, все они размещены в Интернете, можешь поискать и выбрать тот, который понравится именно тебе, но мы сейчас рассмотрим один из наиболее простых, который подходит для наших нужд.

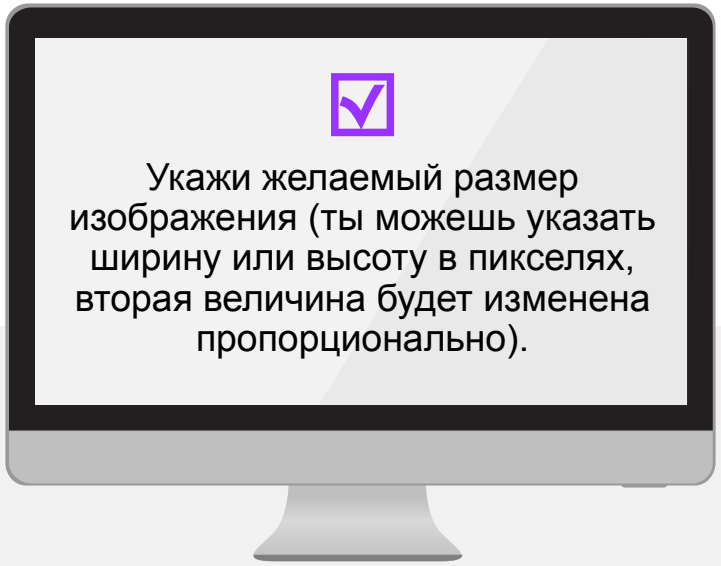
Ты найдёшь этот конвертер по адресу:

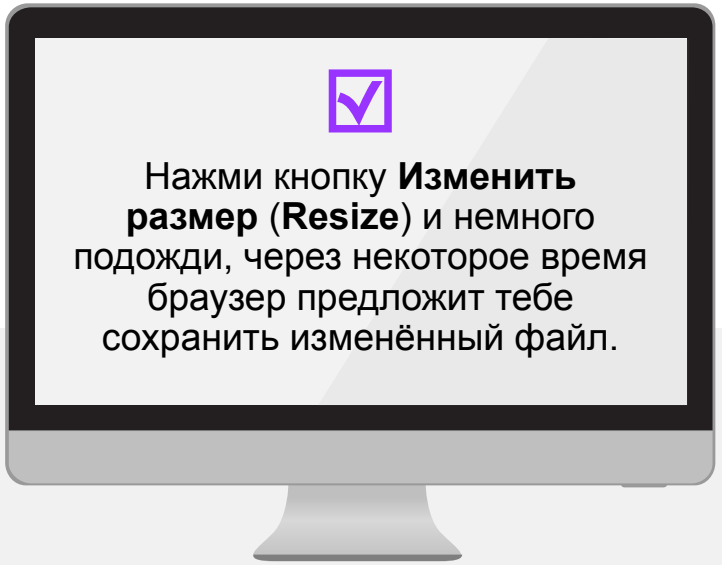


Сохрани адрес этого сайта себе в Закладки, возможно, он тебе ещё пригодится!

ИНСТРУКЦИЯ ДЛЯ ДАЛЬНЕЙШИХ ДЕЙСТВИЙ ПРИВЕДЕНА ПРЯМО НА ОТКРЫВШЕЙСЯ СТРАНИЦЕ.



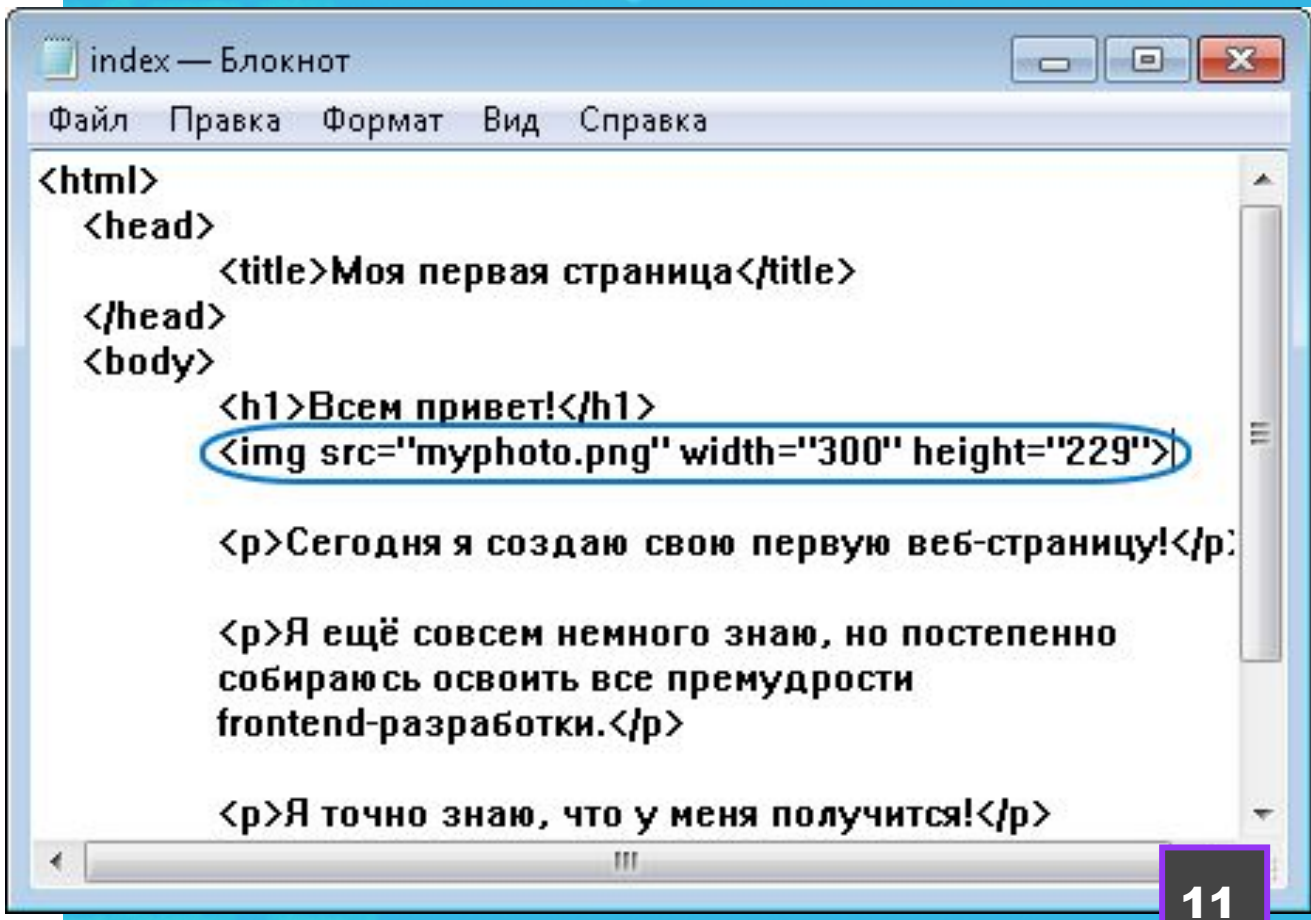


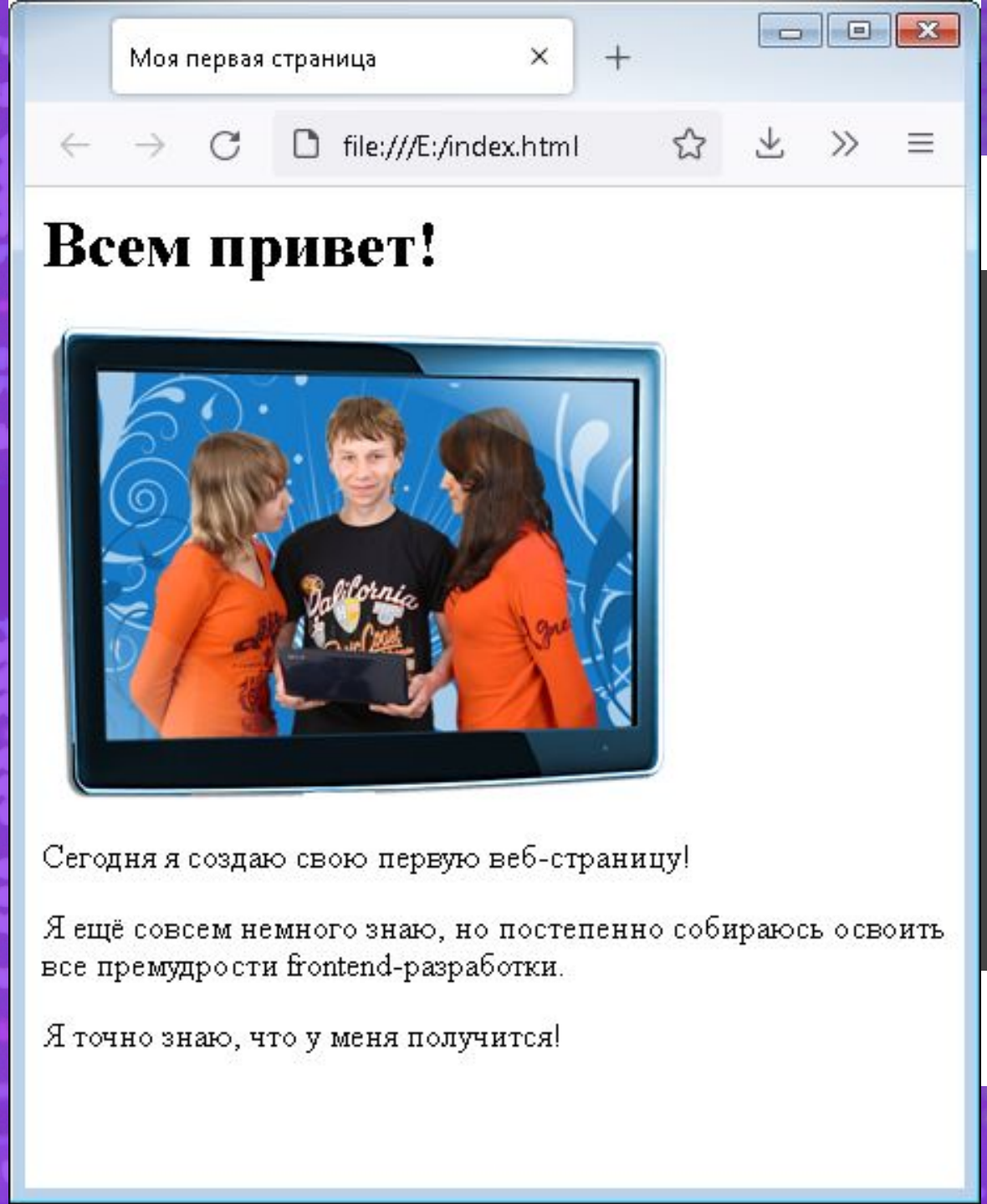


ВОТ И ВСЁ! И НЕ НУЖНО РАЗБИРАТЬСЯ ИНТЕРФЕЙСОМ ГРАФИЧЕСКОГО РЕДАКТОРА.

Но вот если тебе нужно откадрировать изображение, то есть не просто изменить его размеры, а вырезать из него некоторую область (например, сделать портрет из фотографии в полный рост), то без графического редактора никак не обойтись.

- Сохрани или скопируй уменьшенную фотографию в то место, где размещена сейчас твоя страница.
- Переименуй файл изображения, например, назови его myphoto.
- Запиши HTML-код вставки изображения на страницу сразу после заголовка, не забудь указать реальные размеры именно твоего изображения (изображение может выглядеть искажённым, если ты не сохранишь правильное соотношение сторон фотографии, поэтому если ты точно знаешь только высоту или только ширину, можешь указать всего один атрибут):





При записи атрибута src обрати внимание на расширение файла. В нашем примере расширение .png означает, что это изображение сохранено в формате PNG.

Твоя фотография, скорее всего, будет иметь расширение .jpg, так как наиболее популярным форматом для хранения фотографий является формат JPEG, но к графическим форматам мы вернёмся чуть позже.

Сохрани внесённые в файл изменения и посмотри на страницу в браузере**,** должно получиться примерно так **(**только с твоей фотографией**):**

А ТЕПЕРЬ ПРОВЕДЁМ МАЛЕНЬКИЙ ЭКСПЕРИМЕНТ!

- Допиши в теге <img> атрибут alt, указав в нём альтернативный текст, например:

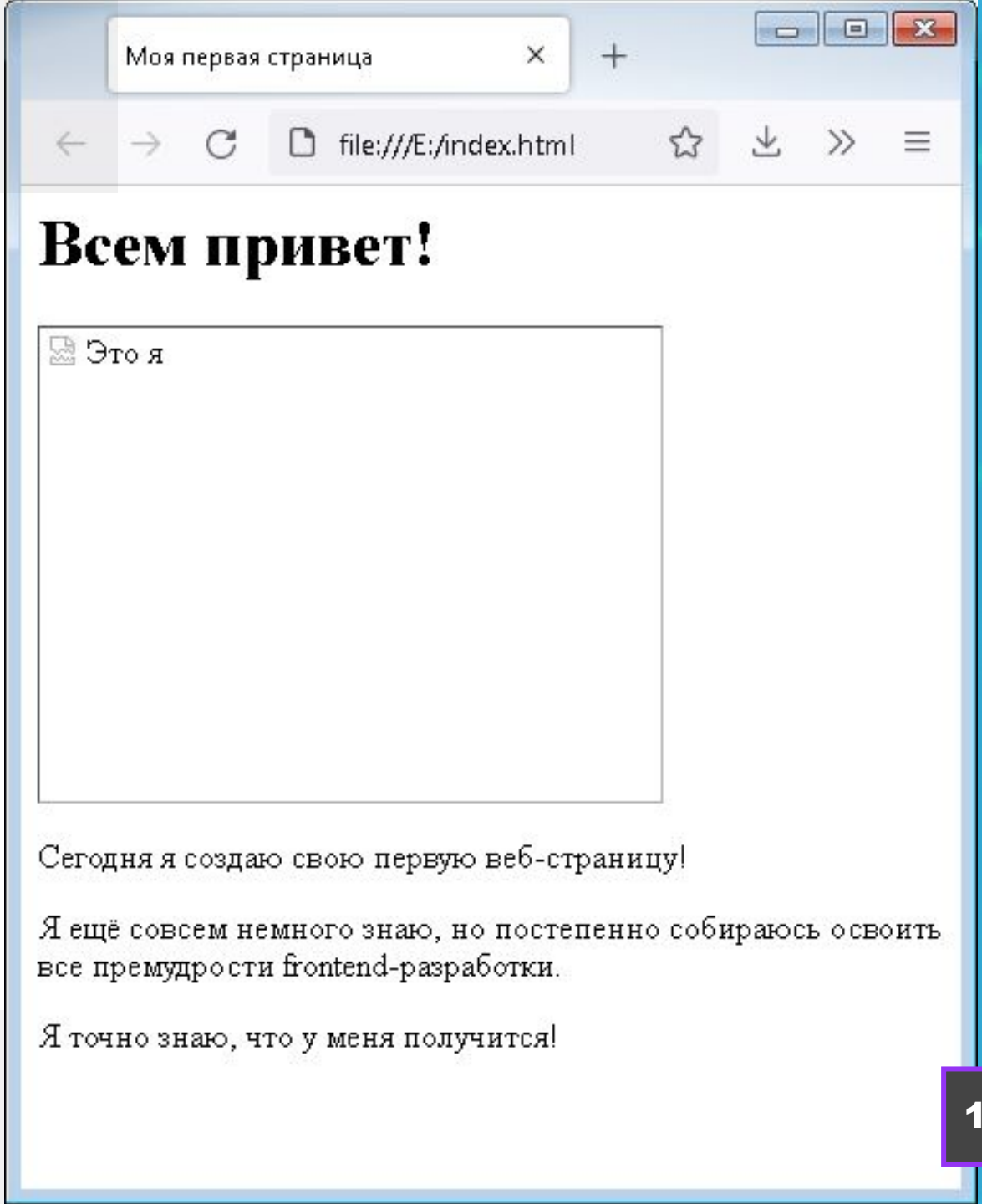
<img src="myphoto.png" width="300" height="229" alt="Это я">

- Сохрани изменения и скопируй файл index.html в какую-нибудь другую папку, не копируя при этом файл самого изображения.

- Запусти файл в браузере из его нового места размещения:

- Так как файл изображения теперь для страницы недоступен, вместо фотографии на странице будет присутствовать зарезервированное под её размеры место, а также будет отображён альтернативный текст, поясняющий, что изображено на отсутствующей картинке.

Эксперимент завершён, поэтому копию файла **index.html** теперь можно удалить.



ВЕРНЁМСЯ К РАЗГОВОРУ О ФОРМАТАХ ГРАФИЧЕСКИХ ФАЙЛОВ.

Несмотря на большое разнообразие графических форматов, в веб-разработке чаще всего используются: JPEG, GIF, PNG.

Почему именно они**?**

Каждый из этих форматов позволяет не только редактировать изображение в любом графическом редакторе, но и сжимать его, уменьшая размеры, тем самым увеличивая скорость загрузки изображения на страницу. Рассмотрим особенности каждого из них.

JPEG (Joint Photographic Experts Group) — данный формат является наиболее популярным для хранения фотографий и фотореалистичных изображений. Этот формат использует метод сжатия с потерями, поэтому при работе с ним иногда приходится выбирать между размером файла и качеством изображения.

ВЕРНЁМСЯ К РАЗГОВОРУ О ФОРМАТАХ ГРАФИЧЕСКИХ ФАЙЛОВ.

GIF (Graphics Interchange Format) — этот формат был создан для преобразования художественных, текстовых и одноцветных изображений в очень маленькие файлы, доступные для передачи при модемном соединении. Максимальное число цветов в изображении этого формата не может превышать 256, что намного меньше, чем нужно для фотографии даже с низким разрешением. Поэтому фотографии обычно не преобразовывают в формат GIF.

Зато формат GIF позволяет создавать анимированные изображения. А ещё один плюс этого формата — поддержка прозрачности: если нарисовать рисунок на прозрачном фоне, а затем поместить его на страницу, то будет создаваться ощущение, что рисунок нарисован прямо на ней, так как через прозрачный фон картинки будет виден фон страницы.

Но с прозрачностью в GIF нужно обращаться осторожно, он не поддерживает полупрозрачность, то есть отображает только полностью прозрачные пиксели либо полностью непрозрачные, из-за этого, если преобразовать в GIF изображение с полупрозрачными переходами и тенями, можно увидеть вместо них «рваные» края.

ВЕРНЁМСЯ К РАЗГОВОРУ О ФОРМАТАХ ГРАФИЧЕСКИХ ФАЙЛОВ.

PNG (Portable Network Graphics) — этот формат использует метод сжатия без потерь, поэтому фотографии в этом формате не сильно отличаются от изображений, сохранённых в формате JPEG, при этом файл PNG может на самом деле иметь намного меньший размер.
К тому же формат PNG поддерживает не только прозрачность, но и полупрозрачность, чего не позволяет формат GIF.

Таким образом, формат JPEG лучше всего подходит для фотографий, формат GIF — для рисунков, ну а формат PNG оптимален и для того, и для другого.

Перечисленные форматы являются форматами растровых изображений. Файл растрового изображения содержит информацию о расположении и цвете каждого пикселя, поэтому для качественного отображения такого файла на веб-странице необходимо, чтобы реальный размер изображения совпадал с указанными в атрибутах width и height размерами в пикселях.

ЧТОБЫ УБЕДИТЬСЯ В ЭТОМ, ПРОВЕДЁМ ЕЩЁ ОДИН ЭКСПЕРИМЕНТ.

- Открой файл **index.html** и запиши в атрибутах **width** и **height** числа, в три раза превышающие исходные размеры твоего изображения.

- Сохрани внесённые изменения и посмотри на обновлённую страницу в браузере.

- Наверняка твоя фотография теперь выглядит зернистой или размытой, как будто она плохого качества.

- Как уже было сказано, растровое изображение содержит информацию о положении и цвете каждого пикселя. При искусственном увеличении размера изображения на веб-странице с помощью HTML-атрибутов, каждый пиксель также увеличивается, охватывая несколько пикселей на экране, поэтому изображение и перестаёт быть чётким.

- Именно поэтому ещё до вставки растровых изображений на веб-страницу рекомендуется использовать графические редакторы или онлайн-конвертеры для подгонки фотографий и рисунков к нужному размеру.

Помимо привычных нам растровых форматов графики, существует также векторный формат SVG.

Для создания SVG-изображений используются редакторы векторной графики, такие как Inkscape или Illustrator.

Файл векторного изображения содержит алгоритм, по которому компьютер может вычислить, как должно выглядеть изображение, когда выводится на экран, а не информацию о каждом пикселе. Благодаря этому файлы векторных изображений намного меньше растровых по размеру и имеют высокую масштабируемость — при увеличении масштаба изображения пиксели не увеличиваются вместе с графикой, а потому векторное изображение при увеличении продолжает выглядеть ровным и красивым.

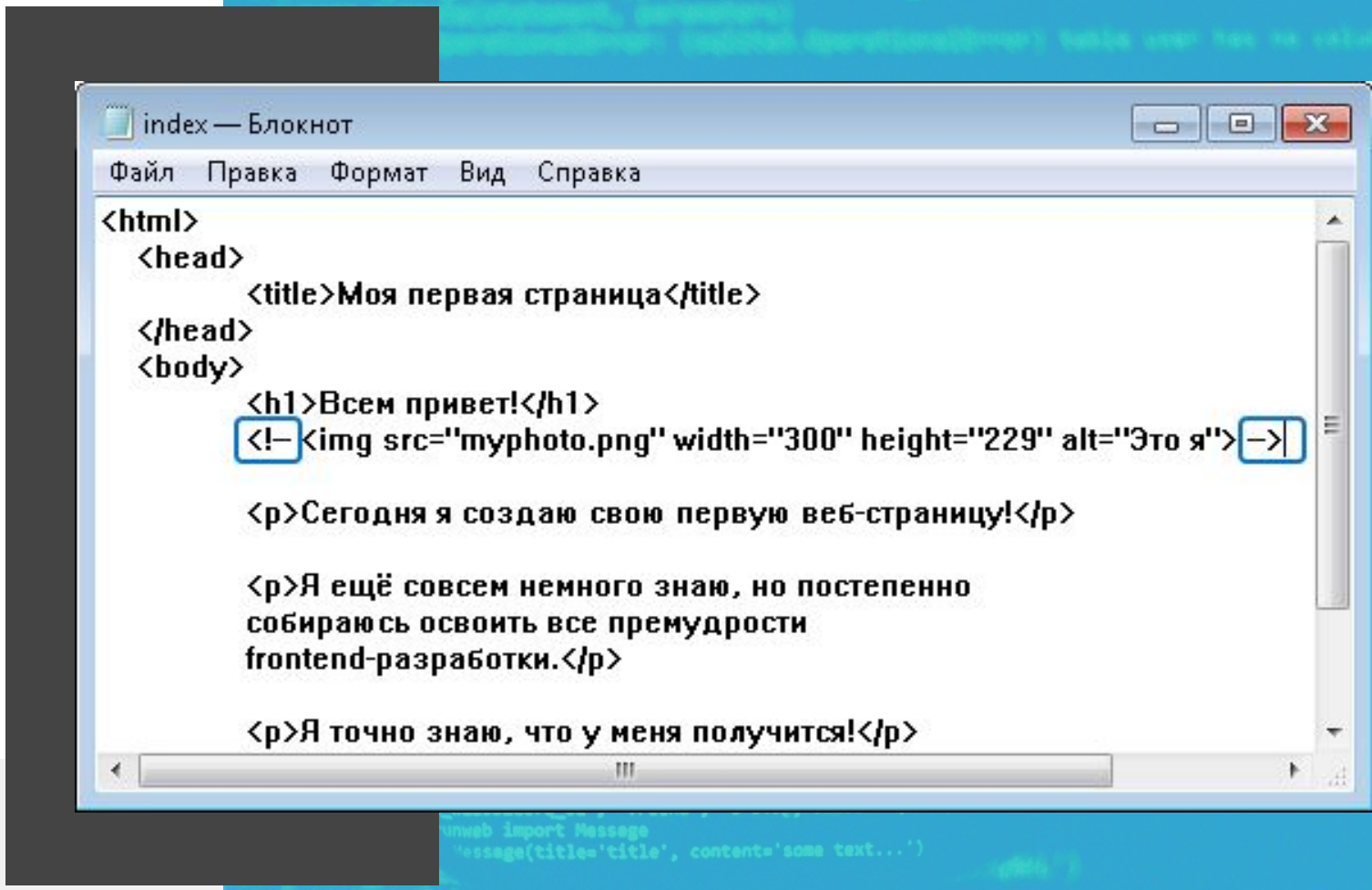
Чтобы убедиться в сказанном, заменим фотографию на нашей странице изображением формата SVG. Для этого тебе потребуется скачать файл smile.svg на свой компьютер и поместить его рядом с файлом index.html.

Вместо удаления кода вставки фотографии на страницу, просто закомментируем его. Комментарий в HTML-коде задаётся так:

<!-- любой текст **-->**

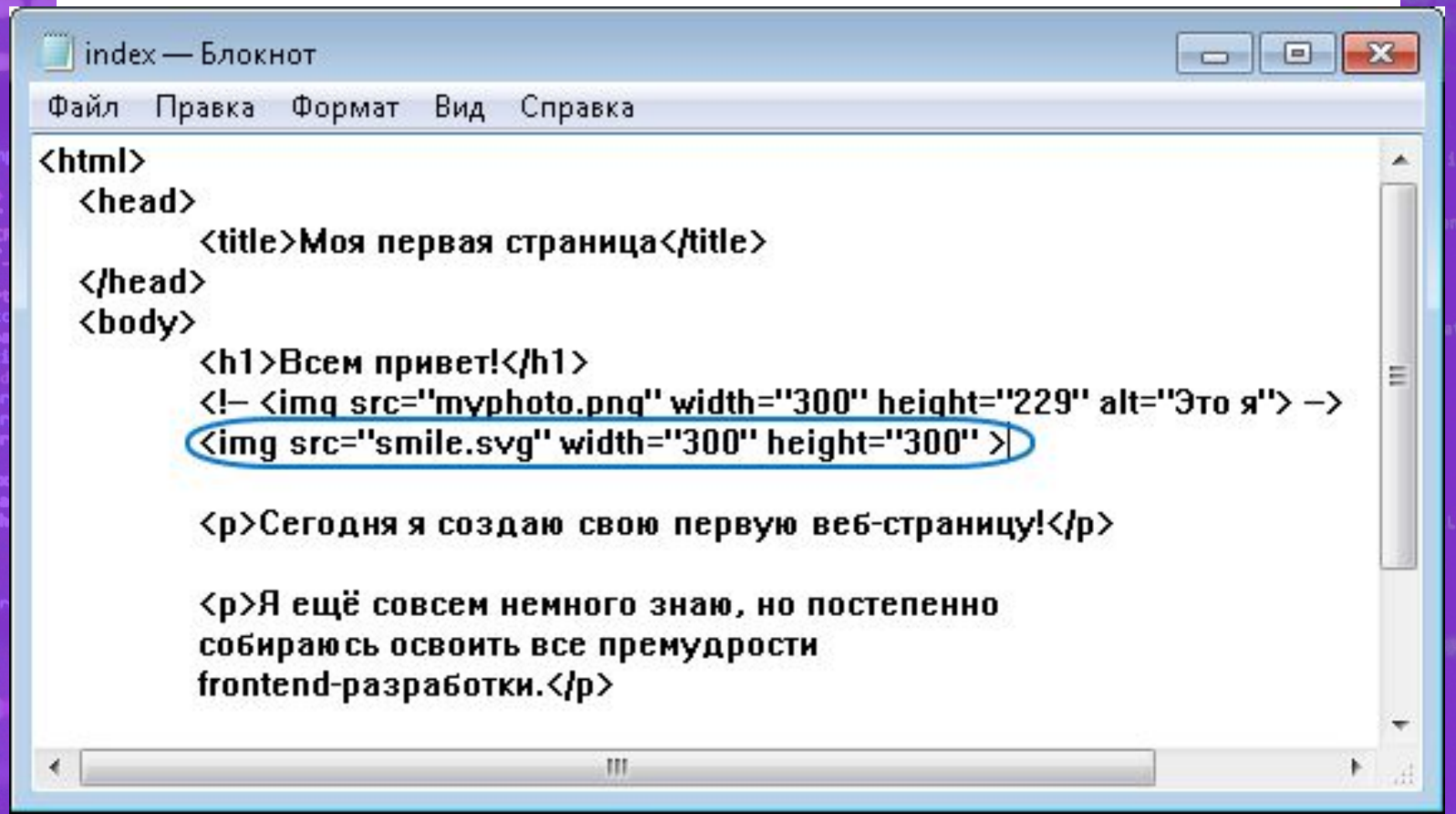
Текст внутри комментария не отображается браузером на странице, поэтому для временного отключения кода достаточно просто поместить его в комментарий.

Закомментируй код вставки изображения**:**



- Сохрани изменения и обнови страницу в браузере, чтобы убедиться, что фотография больше не отображается на странице.

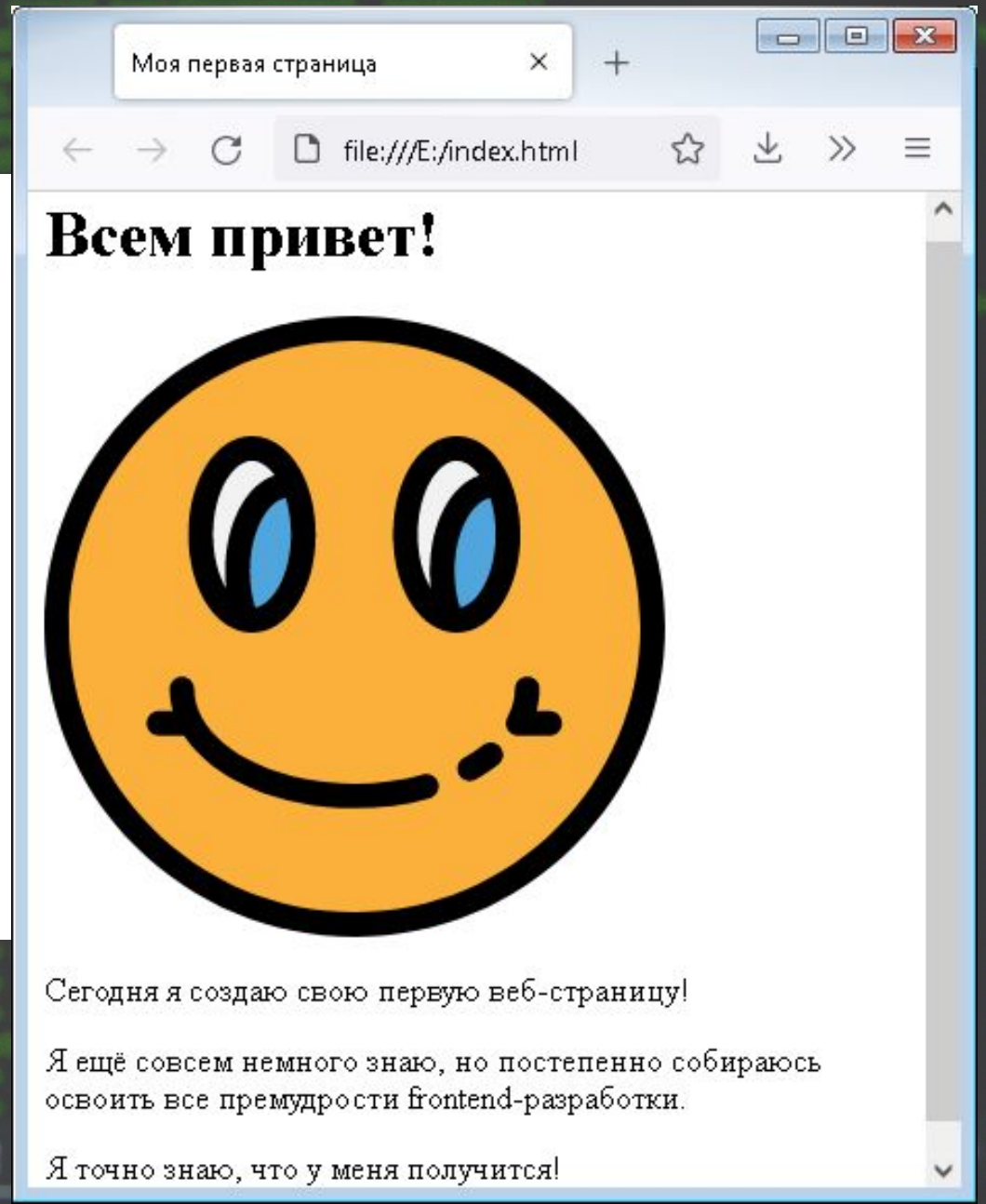
- Допиши код вставки векторного изображения, указав в атрибутах width и height одинаковые размеры, так как данное изображение квадратное:



- Сохрани изменения и обнови страницу в браузере. Теперь вместо фотографии ты увидишь на странице изображение смайла:

- Поэкспериментируй с размером изображения, изменяя ширину и высоту на большие или меньшие значения, и убедись, что при любом размере качество изображения не ухудшается.

Кстати, обрати внимание, если ты будешь изменять только высоту или только ширину, изображение будет подстраиваться под меньший из указанных размеров, а в высоту или в ширину будет растягиваться лишь фон изображения. Этот факт объясняется тем, что файл векторного изображения содержит не сам рисунок, а алгоритм, по которому компьютер вычисляет, как должно выглядеть изображение, а затем выводит его на экран.



Вообще-то, **SVG** — это не просто формат файлов векторной графики, это язык разметки, как и HTML, только предназначен он для разметки графики и содержит элементы для определения фигур, из которых состоит изображение, а также параметров их отображения.

Благодаря этой особенности, ты можешь открыть файл изображения формата SVG при помощи текстового редактора, скопировать фрагмент SVG-кода, который начинается и заканчивается тегами <svg></svg>, и вставить его в HTML-документ. Такой приём иногда называют **встраиванием SVG** непосредственно на веб-страницу.

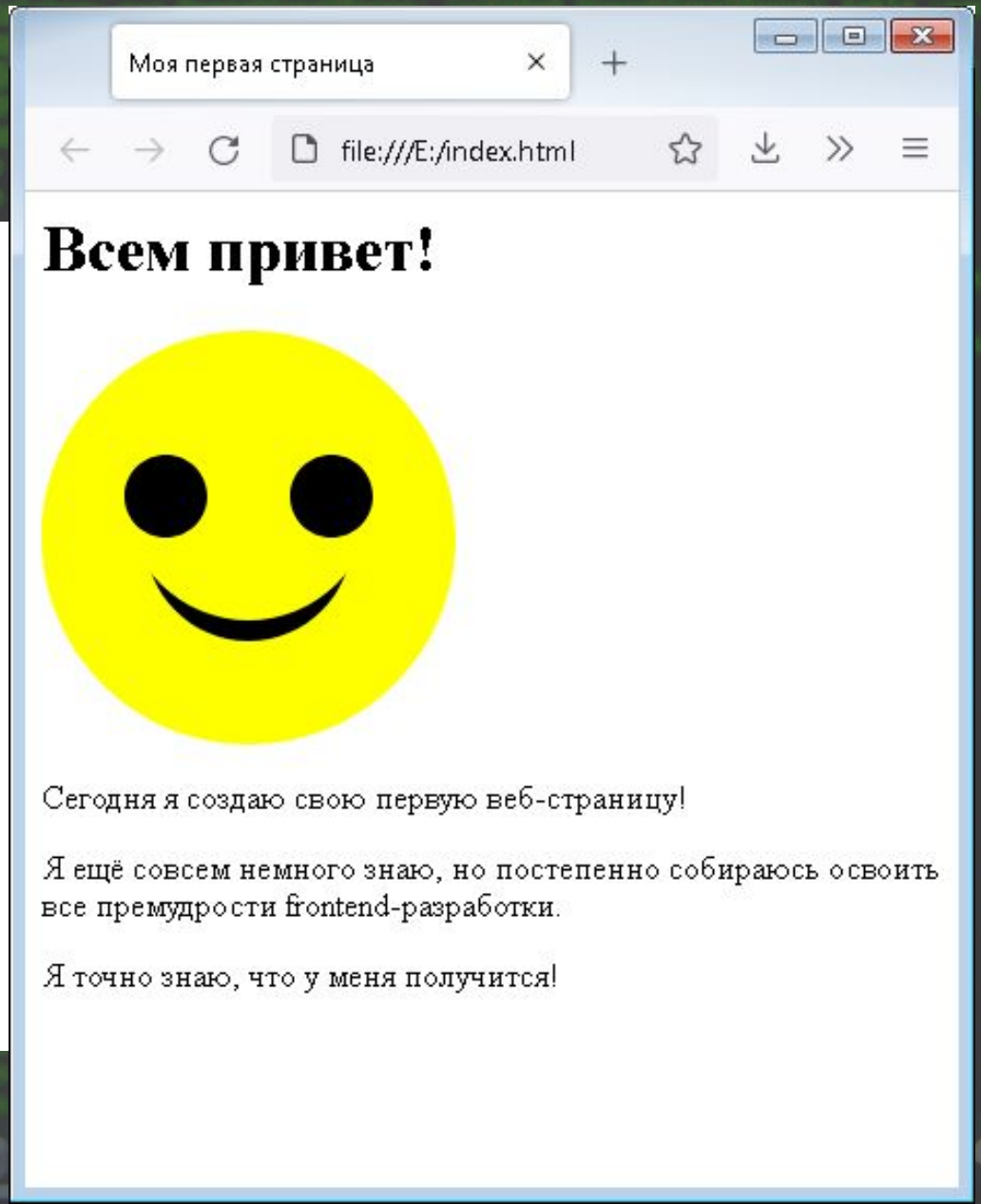
Можешь проделать этот эксперимент, если захочешь, но сложные изображения SVG таким способом на веб-страницу обычно не встраиваются, а внедряются с помощью тега <img>, как было рассмотрено ранее. Зато совсем простые SVG можно легко создавать вручную прямо в текстовом редакторе, не создавая при этом отдельного файла изображения.

Давай попробуем!

- В файле index.html закомментируй ещё и код вставки векторного изображения, а затем помести под ним следующий код:

```
<svg width="200" height="200" viewBox="0 0 200 200">
  <circle cx="100" cy="100" r="100" fill="yellow" />
  <circle cx="100" cy="100" r="50" fill="black" />
  <circle cx="100" cy="80" r="60" fill="yellow" />
  <circle cx="60" cy="80" r="20" fill="black" />
  <circle cx="140" cy="80" r="20" fill="black" />
</svg>
```

Сохрани изменения и обнови страницу в браузере. Теперь ты увидишь на странице изображение смайла, которое вычислено и отрисовано на основе этого кода:



В данном коде задана последовательная отрисовка пяти окружностей с соответствующими координатами центров этих окружностей, радиусами и цветами заливки.

Координаты фигур заданы относительно координат, указанных в атрибуте **viewBox**, это как бы исходный размер изображения.

А атрибуты width и height определяют, сколько места изображение занимает в браузере.

Поэтому если в теге <svg> просто поменять размеры width и height, то вся картинка перерисуется, новые координаты будут вычислены автоматически.

Обрати внимание на символ / перед закрывающей тег **<circle>** угловой скобкой, он обязателен для всех непарных тегов в SVG!

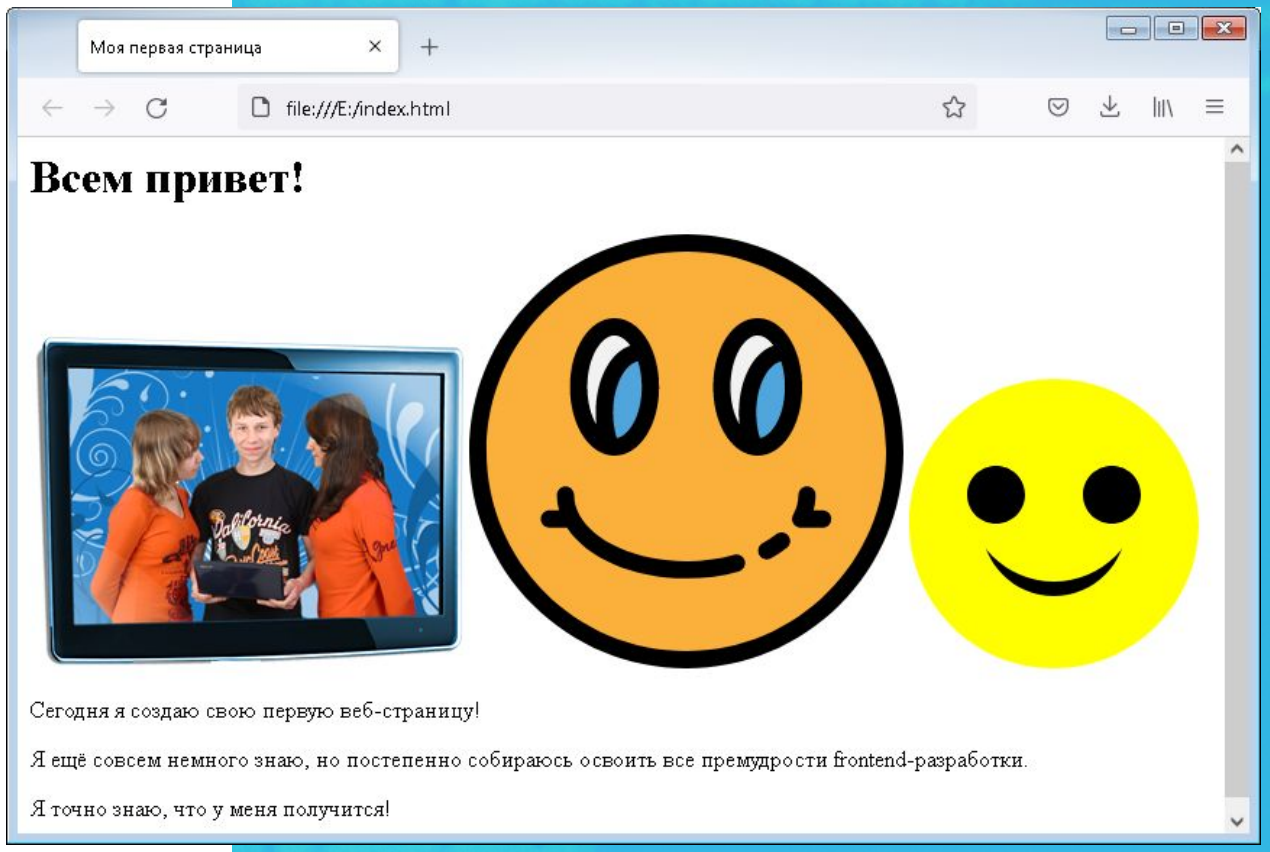
Что ещё следует знать об изображениях на веб-странице?
Изображение — это строчный элемент!

В языке **HTML** существуют блочные и строчные элементы. Браузеры отображают блочные элементы с переводом строки до и после элемента, то есть блочные элементы всегда начинаются с новой строки, а строчные элементы размещаются непосредственно в строке.

Чтобы понять, о чём речь, вернёмся к нашей первой веб-странице.

- Открой в текстовом редакторе файл index.html и убери комментарии, в которые были помещены первые два изображения.

- Открой страницу в браузере и убедись, что все три изображения отображаются в одной строке (они могут переместиться на новую строку, только если ширина окна браузера не позволит им поместиться в одной строке):



index — Блокнот

Файл Правка Формат Вид Справка

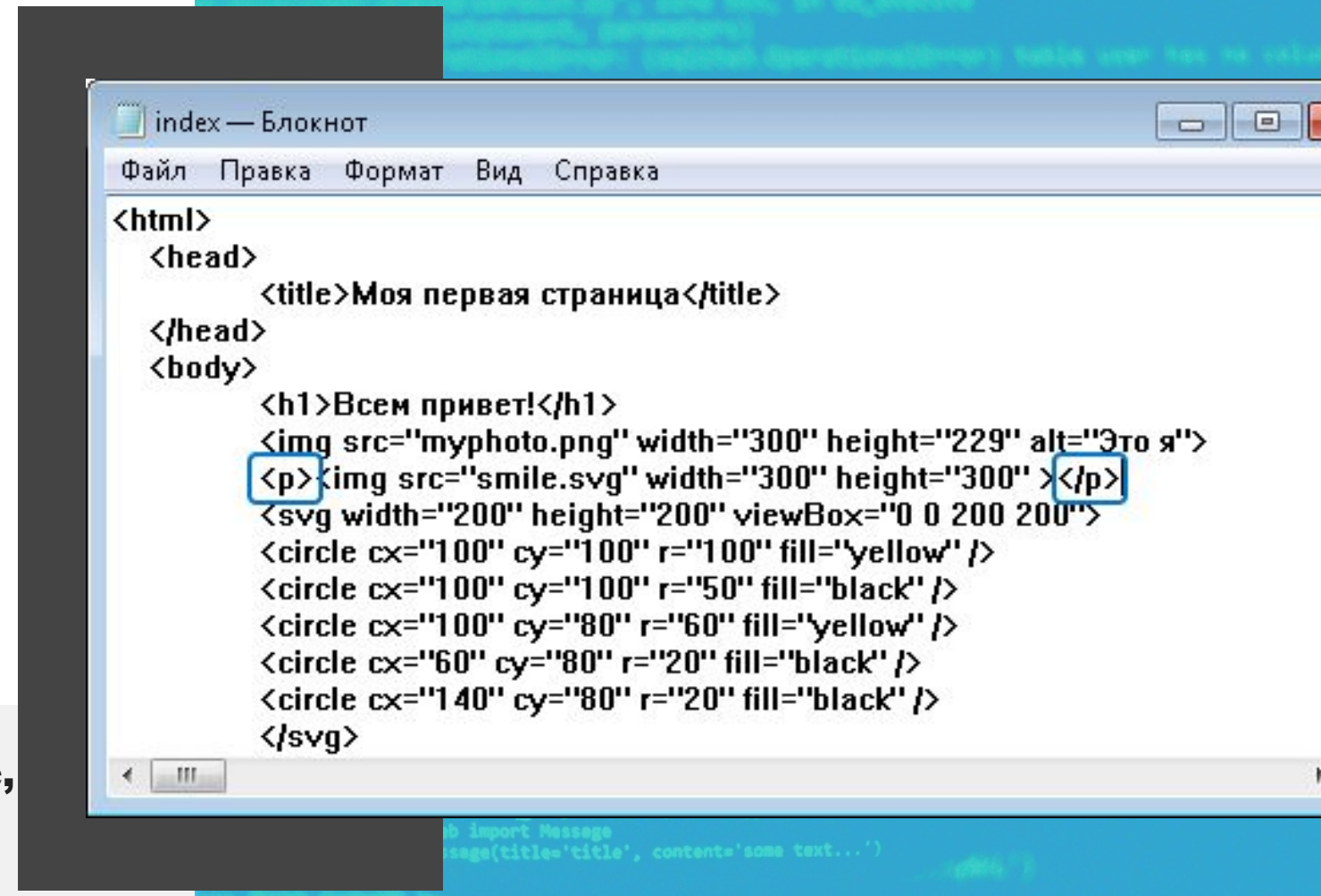
```
<html>
  <head>
        <title>Моя первая страница</title>
  </head>
  <body>
        <h1>Всем привет!</h1>
        <img src="myphoto.png" width="300" height="229" alt="Это я"><br>
        <img src="smile.svg" width="300" height="300" ><br>
        <svg width="200" height="200" viewBox="0 0 200 200">
        <circle cx="100" cy="100" r="100" fill="yellow" />
        <circle cx="100" cy="100" r="50" fill="black" />
        <circle cx="100" cy="80" r="60" fill="yellow" />
        <circle cx="60" cy="80" r="20" fill="black" />
        <circle cx="140" cy="80" r="20" fill="black" />
        </svg>
```

Для того чтобы разместить каждое изображение на отдельной строке, придётся воспользоваться тегом принудительного переноса строки
 или поместить изображения внутрь блочных элементов. К блочным элементам относятся, например, абзац ( ) или заголовок (<h1></h1>).
Проверим оба способа, чтобы понять, как это работает.

- Допиши теги
 после первого и второго тегов <img>:

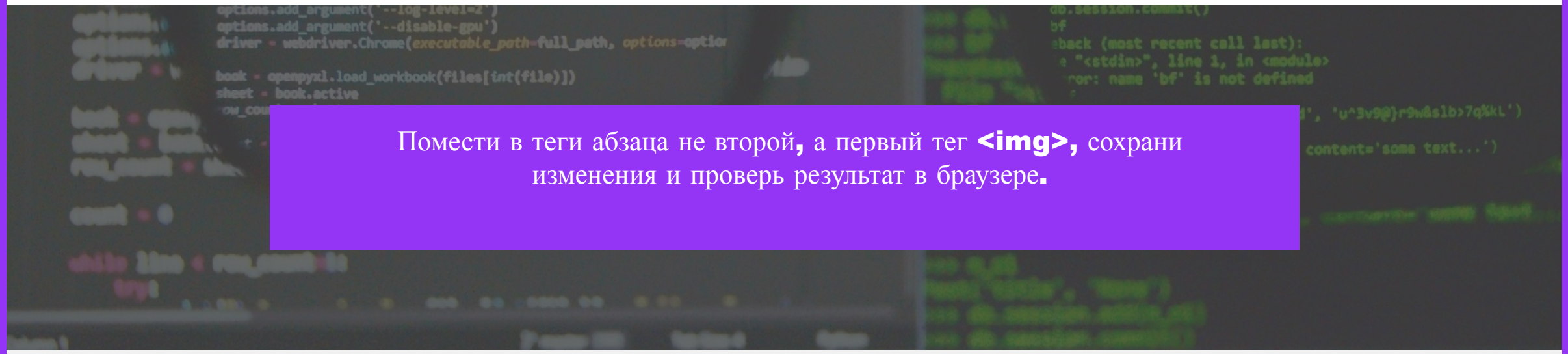
Обрати внимание, что вставлять тег
 после третьего изображения, которое внедрено на страницу с помощью тегов <svg></svg>, мы не стали. Но это не потому, что элемент <svg></svg> блочный (он так же, как и элемент <img>, является строчным), а потому, что далее в коде идёт блочный элемент , его содержимое браузер автоматически отобразит с новой строки.

- Сохрани изменения и обнови страницу в браузере, чтобы убедиться, что каждое изображение теперь размещается на новой строке.
- Теперь удали теги
 и помести второе изображение в теги :



Сохрани изменения и обнови страницу в браузере, чтобы убедиться, что каждое изображение по-прежнему размещается на новой строке.

Поместить в отдельный абзац можно было и каждое изображение, но в данном примере важно понять следующее: если один строчный элемент размещён между двумя блочными элементами, то он тоже будет отображаться на новой строке, так как перевод строки производится и до, и после блочного элемента; если же в коде будет записано несколько строчных элементов подряд, то все они будут отображаться в одну строку.



- Если ты всё сделал правильно, то оба смайла теперь будут отображаться в одной строке.
- На этом эксперименты с нашей первой веб-страницей мы завершаем. Можешь выбрать то изображение, которое хочешь оставить на странице, а остальные два закомментировать. Затем не забудь сохранить файл перед его закрытием.

Далее мы рассмотрим очень интересную возможность **HTML:** научимся накладывать на изображения карты с активными областями**!**

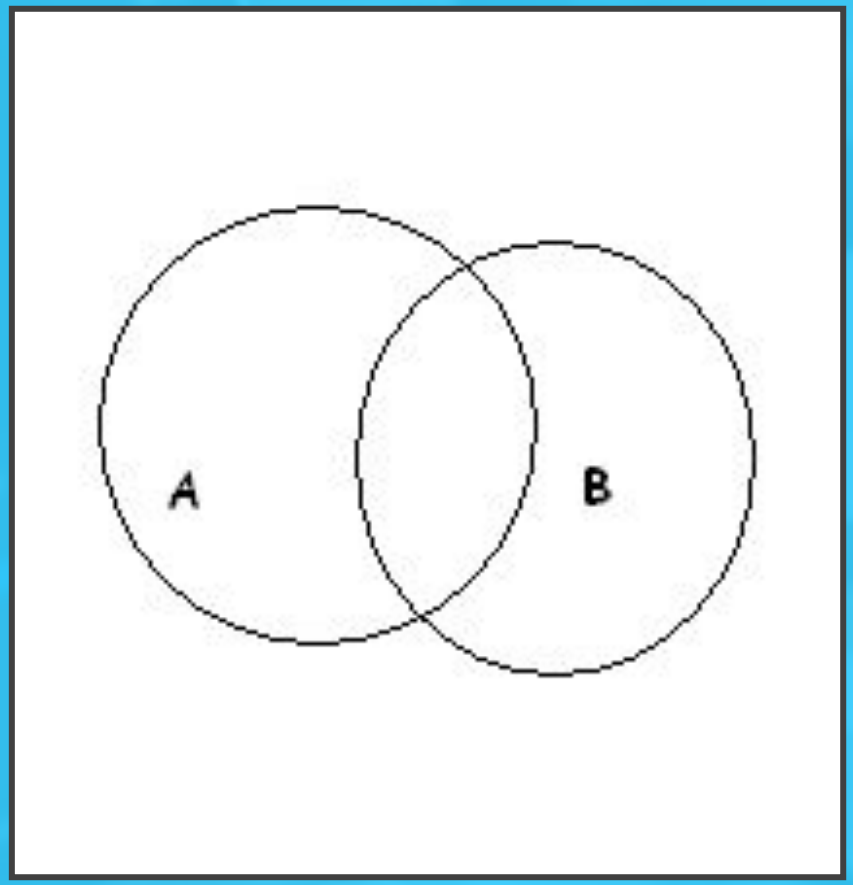
Активные области определяются по изменению вида курсора мыши при наведении на изображение. Щёлкая мышью на активных областях изображения, пользователь может переходить к связанным документам. Таким образом, мы можем сделать из изображения интерактивный элемент, который умеет реагировать на щелчок мыши!

Для изучения этой возможности на реальном примере создадим новый проект**.**

- Создай новую папку (каждый проект лучше хранить в отдельной директории), а в ней создай новый HTML-документ, назови его test.html.

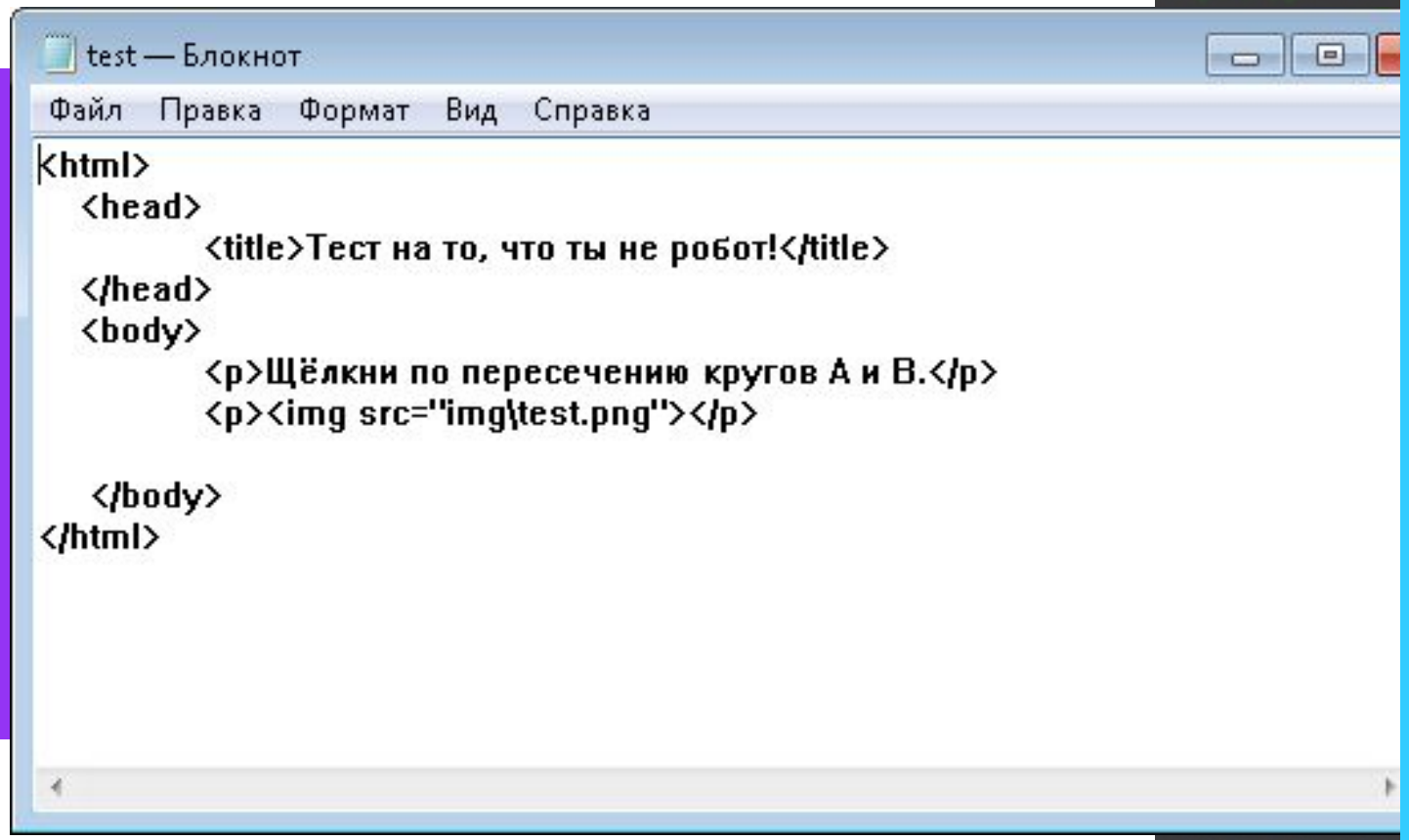
- Создай в директории проекта папку img, скачай и сохрани в ней файл изображения test.png.

Теперь поясним, что именно мы будем делать. Мы создадим простой тест для пользователя: поместим на веб-страницу изображение двух пересекающихся кругов и попросим щёлкнуть в области их пересечения. При этом, наложив на изображение карту активных областей, мы сможем определить, по какой именно области пользователь щёлкнул!



Приступим!

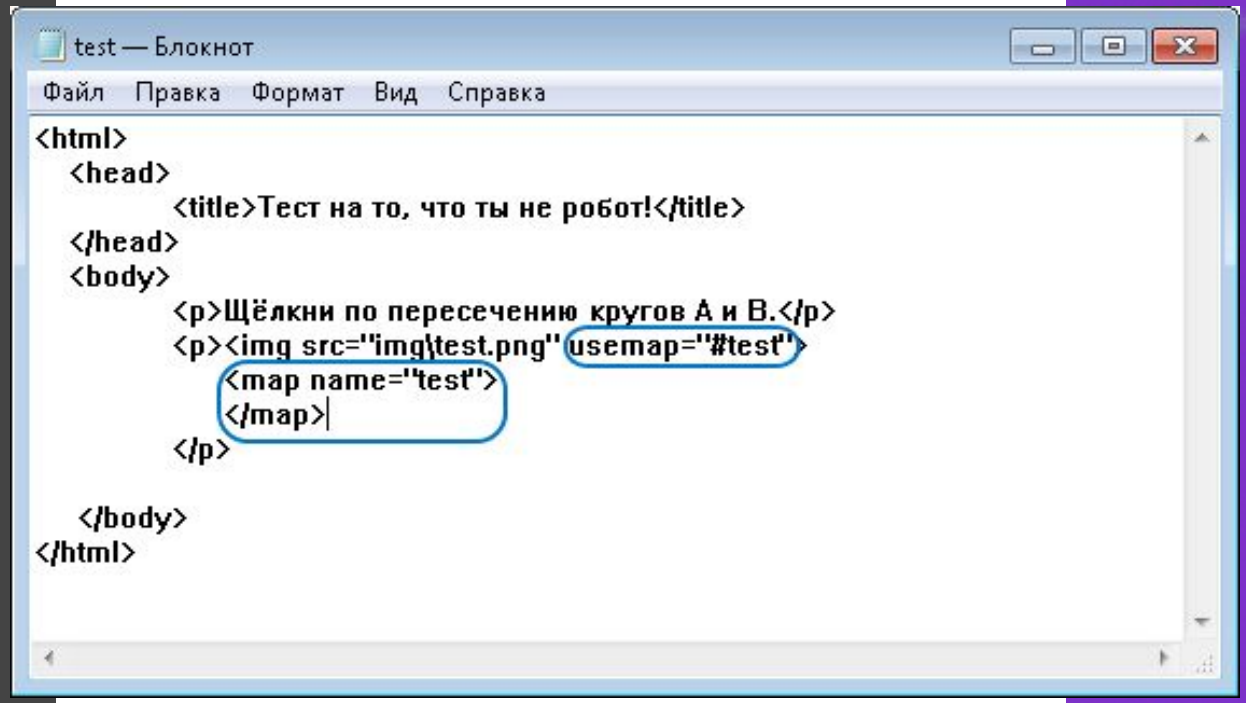
- Открой файл test.html в текстовом редакторе и запиши теги структуры HTML-документа.
- Дай документу оригинальный заголовок, например, **Тест на то, что ты не робот!**.
- Создай на странице два абзаца, в первый помести текст задания, а во второй — изображение:



Наша страница готова, осталось наложить на изображение карту.

Для представления графического изображения в виде карты с активными областями служит элемент <map>.
Для этого элемента доступен атрибут name, который задаёт имя карты.
Значение атрибута name для элемента <map> должно соответствовать имени в атрибуте usemap элемента <img>.

Запиши соответствующие теги и атрибуты в своём коде:



Записав открывающий и закрывающий теги <map></map>, мы создали элемент карты, при помощи атрибута name задали карте имя, а при помощи атрибута usemap в теге <img> указали, что на данное изображение наложена карта с указанным именем.

Обрати внимание**,** что перед именем карты в атрибуте **usemap** обязательно ставится знак решётки **#.**

Таким образом, при помощи языка HTML мы указали браузеру, что на изображение наложена карта, но пока мы ещё не разметили активные области на этой карте, поэтому даже после сохранения изменений в файле, вид курсора на веб-странице при наведении на изображение никак не изменяется.

Внутри элемент **<map>** должен содержать ряд элементов **<area>,** определяющих интерактивные области на изображении**.**

Один элемент **<area>** описывает только одну активную область в составе карты изображения**.**
Элемент **<area>** является непарным тегом и может содержать следующие атрибуты**:**

Атрибут	Описание
alt	Задаёт альтернативный текст для активной области карты
coords	Определяет позицию области на экране. Координаты задаются в единицах длины и разделяются запятыми: для круга — координаты центра и радиус круга; для прямоугольника — координаты верхнего левого и правого нижнего углов; для многоугольника — координаты вершин многоугольника в нужном порядке, также рекомендуется указывать последние координаты, равные первым, для логического завершения фигуры
href	Указывает URL-адрес для ссылки на связанный документ. Может быть указан абсолютный или относительный адрес ссылки
shape	Задаёт форму активной области на карте и формат её координат. Может принимать следующие значения: rect — активная область прямоугольной формы; circle — активная область в форме круга; poly — активная область в форме многоугольника; default — активная область занимает всю площадь изображения
target	Указывает, куда будет загружен документ при переходе по ссылке. Принимает следующие значения: _self — страница загружается в текущее окно; _blank — страница открывается в новой вкладке браузера; _parent — страница загружается во фрейм-родитель; _top — страница загружается в полное окно браузера

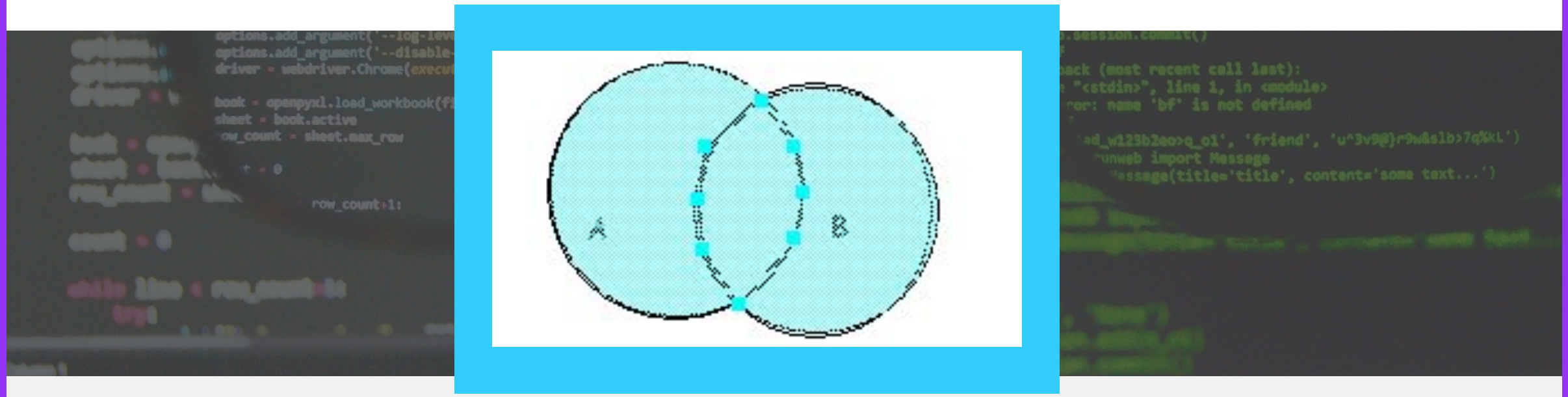
Итак, чтобы создать активную область на карте, нужно поместить в теги <map></map> новый тег <area> и указать в атрибутах форму и координаты активной области, а также ссылку, по которой будет происходить переход при щелчке по данной активной области.

Вычислять координаты активных областей без дополнительных инструментов непросто, но если открыть изображение в любом графическом редакторе и навести указатель мыши на нужную точку, то в строке состояния графического редактора (обычно она находится внизу окна программы) можно увидеть текущие координаты позиции курсора.

Для нашего примера формы и координаты активных областей будут следующими:

```
<area shape="poly"
coords="123,40,137,60,141,80,137,100,11
3,129,97,105,95,83,98,60,123,40" >
<area shape="circle" coords="147,88,54" >
<area shape="circle" coords="86,79,56" >
```

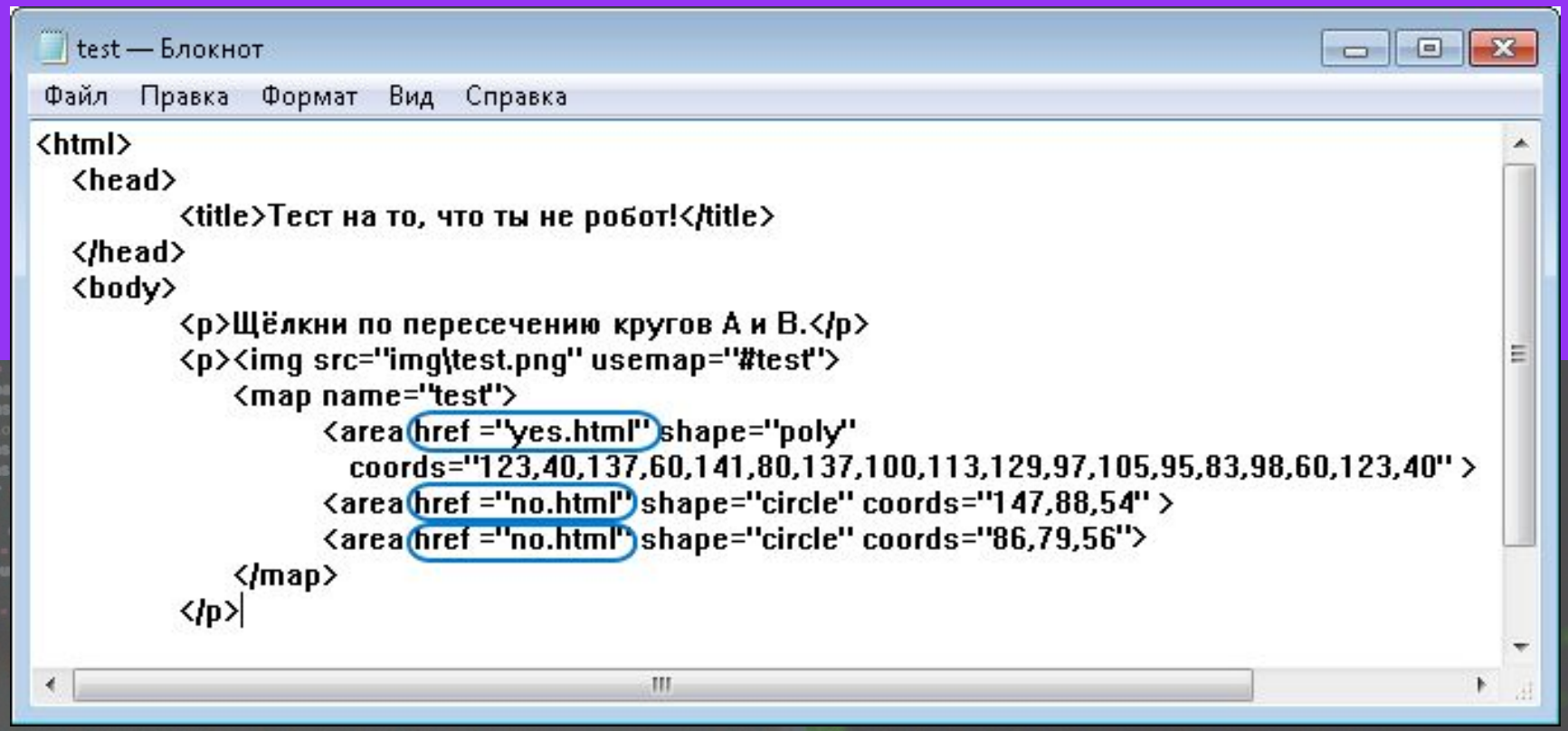
Здесь заданы две активных области, совпадающие с кругами на изображении, а также многоугольник, образующийся на их пересечении. Если бы форма и координаты активных областей были нам видны, то это выглядело бы примерно так (квадратными маркерами выделены координаты точек, из которых складывается многоугольник пересечения):



Обрати внимание, если одна активная область перекрывает другую, то будет работать первая из списка областей, записанных в тегах <map></map>, поэтому область в форме многоугольника на пересечении кругов должна быть записана в коде первой!

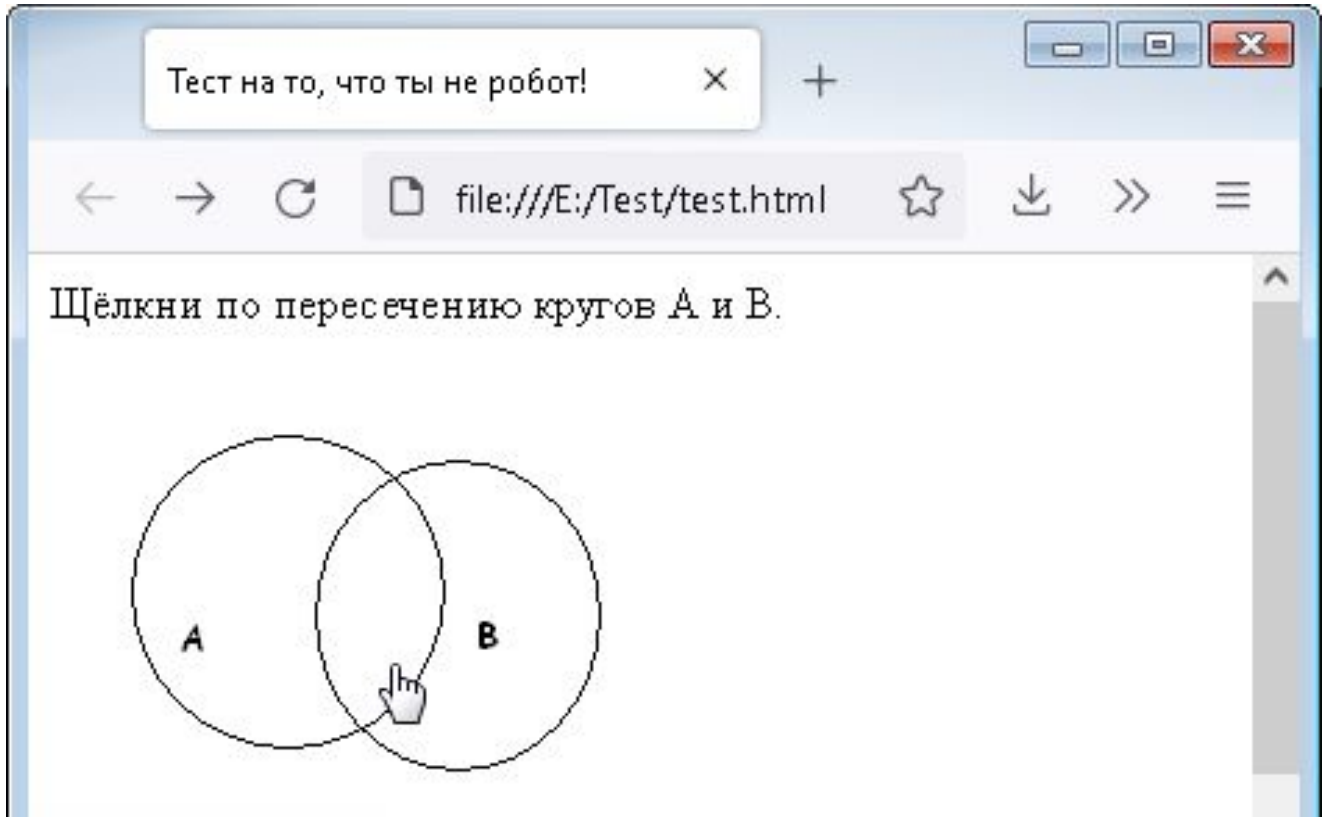
- Запиши в тегах <map></map> соответствующие области <area> и допиши в них атрибуты href со ссылками на страницы yes.html и no.html:

Обязательным также является атрибут, задающий альтернативный текст, но так как текст в атрибуте alt отображается только тогда, когда изображение недоступно, а в нашем примере смысл задания при этом совершенно теряется, то мы можем опустить этот атрибут или записать alt="".



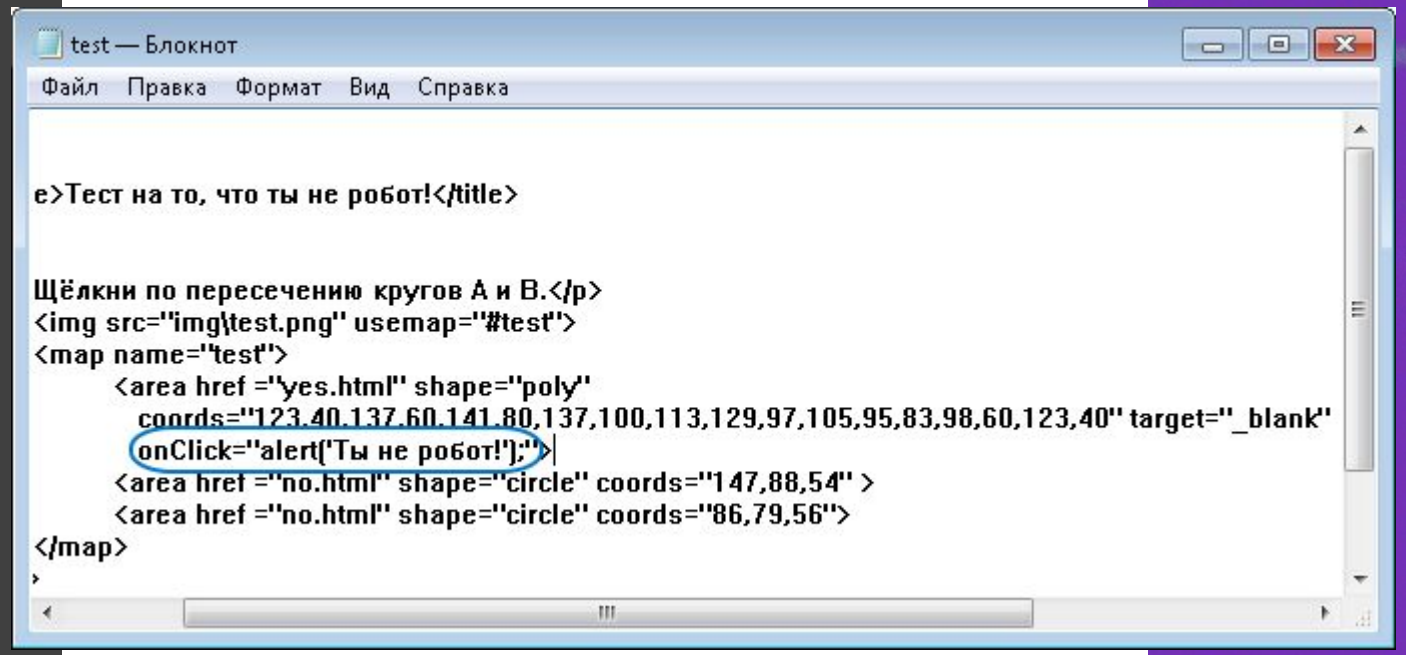
Обязательным также является атрибут, задающий альтернативный текст, но так как текст в атрибуте alt отображается только тогда, когда изображение недоступно, а в нашем примере смысл задания при этом совершенно теряется, то мы можем опустить этот атрибут или записать **alt=""**.

- Сохрани изменения в документе и обнови страницу в браузере. Теперь при наведении мыши на изображение ты увидишь, что курсор изменяет свой внешний вид, как бы приглашая кликнуть:

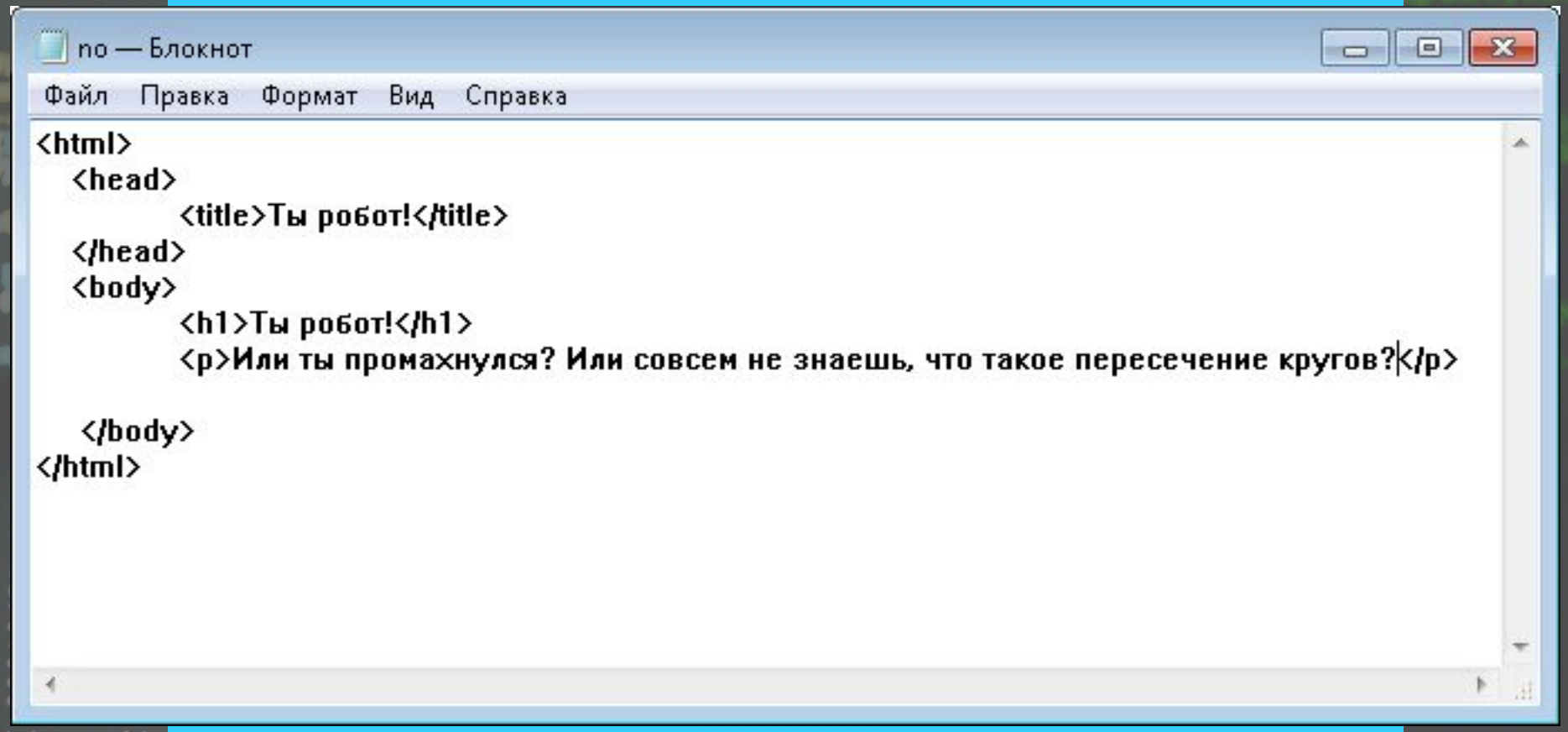


В атрибутах href тегов <area> мы указали ссылки на так называемые связанные документы. Это веб-страницы yes.html и no.html, которые теперь связаны с нашей первой страницей. Вот только их ещё не существует, поэтому щелчок по любой из активных областей изображения приведёт к тому, что браузер сообщит нам о том, что страница не найдена.
Создадим эти страницы, чтобы система связанных документов заработала!

- В папке со страницей test.html создай новые файлы yes.html и no.html.
- В коде страницы yes.html запиши реакцию на правильное нажатие:



В коде страницы **no.html** запиши реакцию на промах:

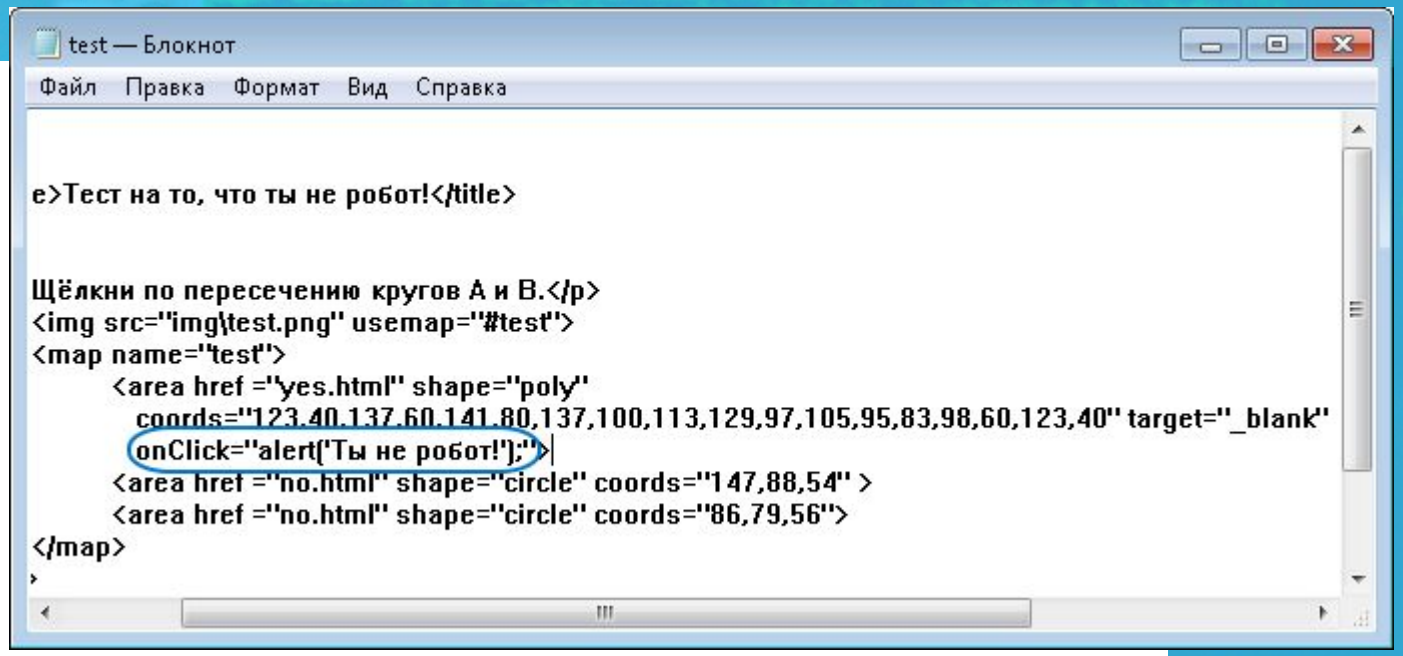


- Сохрани все изменения и протестируй работу теста в браузере.
- Добавь атрибут target в тегах <area>, чтобы настроить открытие связанных с первой страницей документов в новых вкладках браузера.

Использовать карты изображений с активными областями можно не только для перехода по ссылкам к связанным документам. Событие щелчка по активной области изображения можно обработать при помощи языка программирования JavaScript.

Данный язык программирования будет изучаться в отдельном модуле, поэтому сейчас мы не будем реализовывать сложную обработку, а просто поэкспериментируем, чтобы понять, как это работает.

- Допиши в первом теге **<area>** следующий код**:**



- Сохрани изменения и протестируй страницу в браузере.

Теперь перед переходом к связанному документу yes.html появится окно с сообщением «Ты не робот!». Именно так сработает функция alert языка JavaScript, записанная в событии onClick тега <area>.

КОРОТКО О ГЛАВНОМ

- Для вставки изображения на страницу используется тег **<img>:**

<img src="images/logo.jpg">

- Атрибут **src** является обязательным, он указывает браузеру расположение графического файла, который необходимо поместить на страницу.
- Все остальные атрибуты тега **<img>** не являются обязательными и используются по мере необходимости.
- Атрибуты **width** и **height** — ширина и высота изображения в пикселях.
- Атрибут **alt** позволяет задать описание картинки для тех пользователей, у которых в браузере отключён показ графики.
- На изображение можно накладывать карту с активными областями.

КОРОТКО О ГЛАВНОМ

- Для представления графического изображения в виде карты с активными областями служит элемент **<map>.** Для этого элемента доступен атрибут **name**, который задаёт имя карты. Значение атрибута name для элемента <map> должно соответствовать имени в атрибуте **usemap** элемента <img>, при этом перед именем карты в атрибуте usemap обязательно ставится знак решётки **#**.

- Внутри элемент <map> должен содержать ряд элементов <area>, определяющих интерактивные области на изображении.

- Один элемент **<area>** описывает только одну активную область в составе карты изображения.

- Элемент <area> является непарным тегом и обязательно должен содержать атрибуты **shape**, **coords**, **href**, которые задают форму и координаты активной области, а также ссылку, по которой будет происходить переход при щелчке по данной активной области.

ПРАКТИЧЕСКОЕ ЗАДАНИЕ К ТЕМЕ 1.2

ЗАДАНИЕ 1

1

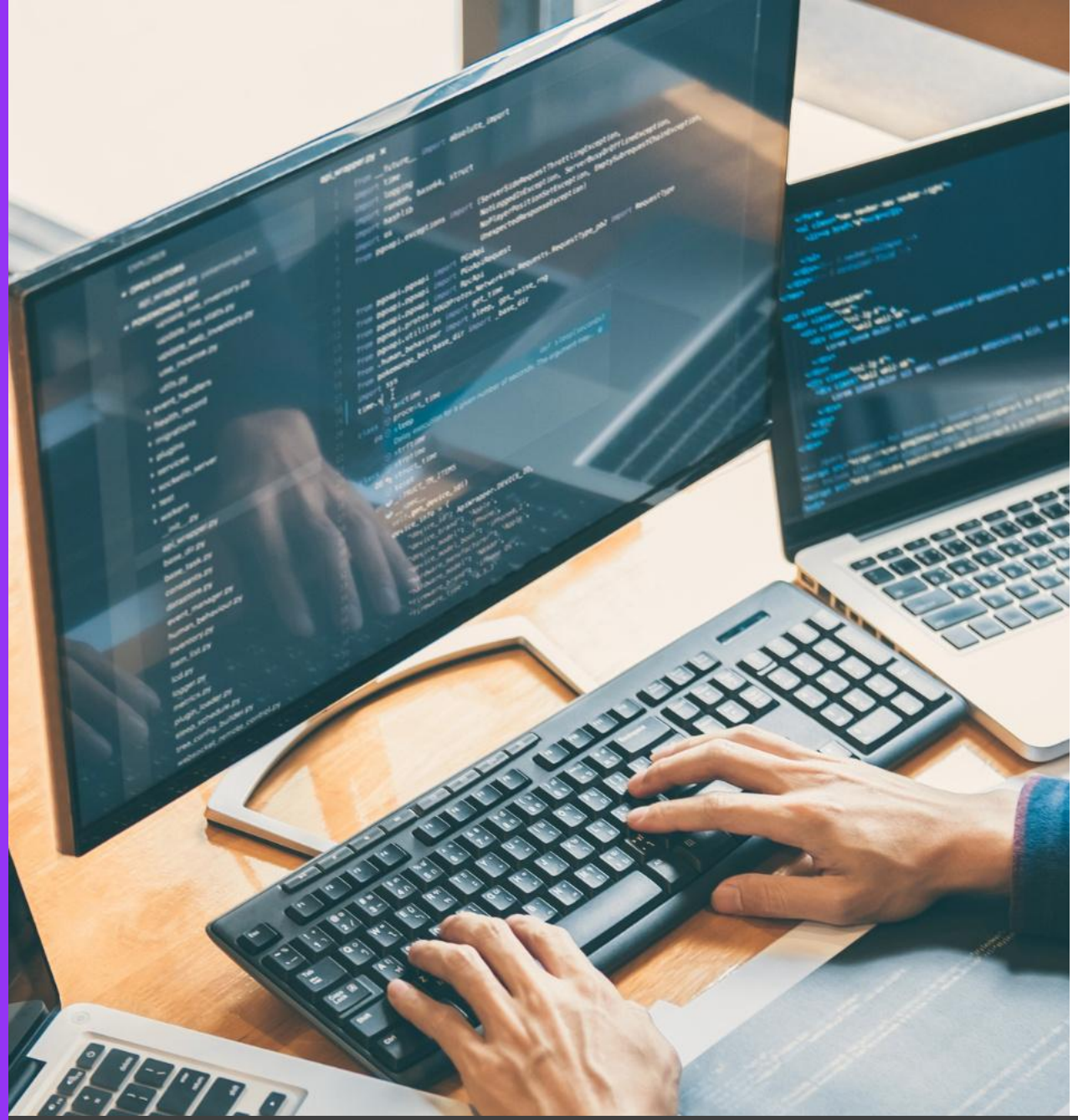
Целью данного курса не является полноценное изучение SVG, но помимо окружностей, для создания примитивных SVG-рисунков можно использовать прямоугольники, многоугольники, эллипсы и линии. **Примеры кода для этих фигур приведены ниже:**

```
<rect x="120" y="80" width="100"
height="100" rx="15" />
<polygon points="100,100 150,25
150,75 200,0" fill="none"
stroke="black" />
<ellipse cx="100" cy="50" rx="100"
ry="50" />
<line x1="0" y1="80" x2="100"
y2="20" stroke="black" />
```

Попробуй поиграть с SVG! Подставляя приведённые примеры в теги <svg></svg>, разберись в параметрах каждой фигуры и создай новую html-страницу со своим собственным уникальным векторным рисунком, состоящим из нескольких фигур. Перед рисунком помести заголовок с его названием.

Учти, что порядок, в котором элементы указаны в коде SVG, определяет порядок их отрисовки. То есть первым будет нарисован элемент, который записан в коде первым, а все последующие будут рисоваться поверх. Так, для следующего рисунка сначала должен быть записан код отрисовки обеих окружностей, а последним отрисуется прямоугольник:

2



ЗАДАНИЕ **2**

Доработай тест, созданный в уроке, следующим образом:

Добавь ещё одну активную область на карту изображения, эта область должна покрывать всю площадь изображения, не занятую кругами.
Добавь новую реакцию на щелчок по пустой части изображения, заголовок и текст для нового файла придумай самостоятельно.
В качестве решения прикрепи к ответу заархивированную папку твоего проекта.

СПАСИБО ЗА ВНИМАНИЕ!
ДО ВСТРЕЧИ НА СЛЕДУЮЩЕМ ЗАНЯТИИ.